FIG.3

START 310

↓

COMPUTE
THE NODE CONFLICT SET
(NCS) 320

↓

COMPUTE THE SCS BY
IDENTIFYING PAIRS OF
CONFLICTING STATEMENTS
IN NCS 330

↓

END 340

FIG.2

START 210

↓

INPUT MULTITHREADED
CONTEXT GRAPH
(MCG) 220

↓

PROCESS MCG TO
DETERMINE
STATEMENT CONFLICT
SET (SCS) 230

↓

OUTPUT SCS 240

↓

PERFORM DYNAMIC
DATARACE DETECTION
ON SCS 250

↓

END 260

FIG.1

100

APPLICATION
PROGRAM 102

OPERATING SYSTEM 108

RAM 114

CPU 116

I/O
INTERFACE 118

112
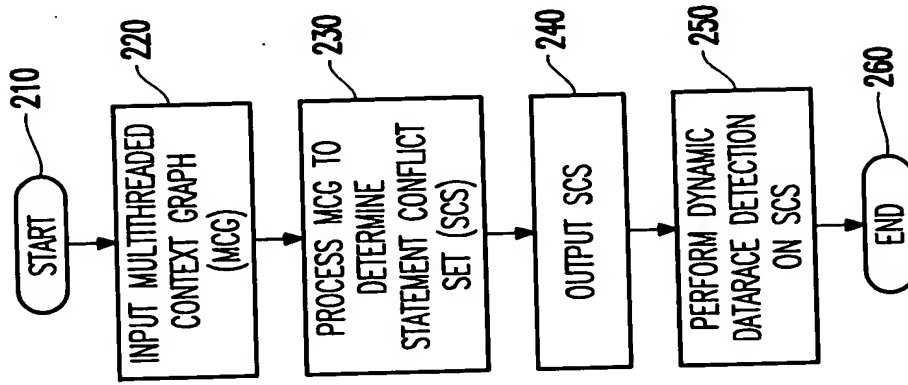
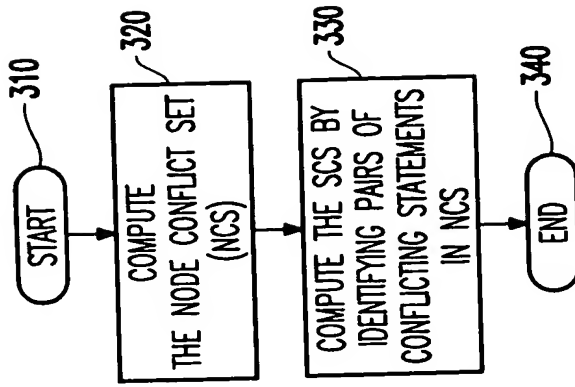TERMINAL 126

DATA
STORAGE 130

PRINTER 134

104

```
// thread T1
sychronized void foo() {
S10:   Obj p, q, r;
       . . . ;
S11:   p.f = ...;
S12:   myRunnable T2 = new myRunnable();
S13:   T2.start(); // start T2

S14:   synchronized(q) {
S15:       p.f = ...;
S16:   } // end of synchronized
S17:   . . . ;
S18:   T2.join();
S19:   . . . ;
} // end of foo()
```

```
// thread T2
public void run () {
S20:   Obj p, q, r;
       . . . ;
S21:   ... = p.f;

S23:   . . . ;
S24:   synchronized(q) {
S25:       ... = p.f;
S26:   } // end of synchronized
S27:   . . . ;
} // end of run()
```
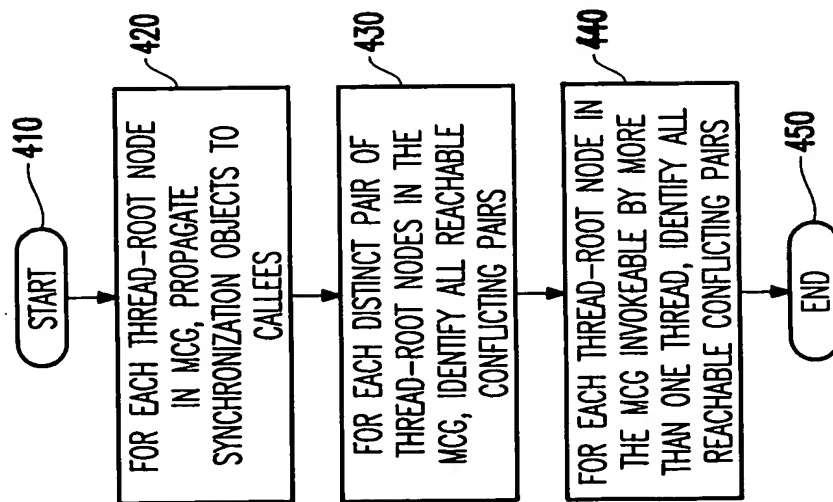
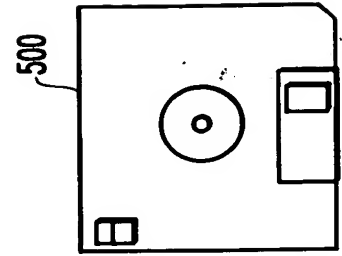EXAMPLE PROGRAM SEGMENT WITH TWO THREADS

FIG.6

FIG.5

START — 410

FOR EACH THREAD-ROOT NODE IN MCG, PROPAGATE SYNCHRONIZATION OBJECTS TO CALLEES — 420

FOR EACH DISTINCT PAIR OF THREAD-ROOT NODES IN THE MCG, IDENTIFY ALL REACHABLE CONFLICTING PAIRS — 430

FOR EACH THREAD-ROOT NODE IN THE MCG INVOKEABLE BY MORE THAN ONE THREAD, IDENTIFY ALL REACHABLE CONFLICTING PAIRS — 440

END — 450

FIG.4